Graph Based Pattern Recognition

Exercise 2

Basis

- Chapters 3 and 4

Submission

- The submission takes place online on ILIAS.
- Solutions to the theory tasks must be submitted as `*.pdf` file. Other formats will not be accepted.
- Source code for the implementation tasks must be submitted as `*.py` files. Source code that cannot be executed will not be accepted.
- Individual submissions or submissions in teams of two are allowed (hand in only one copy per group). In the source code file, include the *names and matriculation numbers* of both group members in the first two lines as comments.

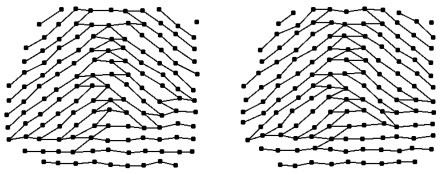
Dates

- Briefing: 15.03.2023
- Submission: 29.03.2023
- Debriefing: 29.03.2023

Theoretical Tasks

1. Consider the graphs illustrated below. Note that in this example each node is labeled with a two-dimensional attribute giving its position in the plane (the graphs represent the ridge structure of two fingerprint images).

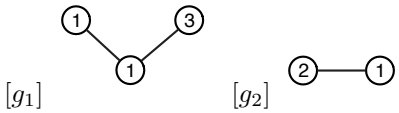
   Elaborate on the problems of exact graph matching paradigms in this particular setting (or vice versa on the benefits of error-tolerant graph matching).



a) b)

2. Define a cost function so that the Graph Edit Distance simulates the (not necessarily induced) subgraph isomorphism, i.e. returns 0 if the first graph is isomorphic to a (not necessarily induced) subgraph of the second graph and ∞ if not.

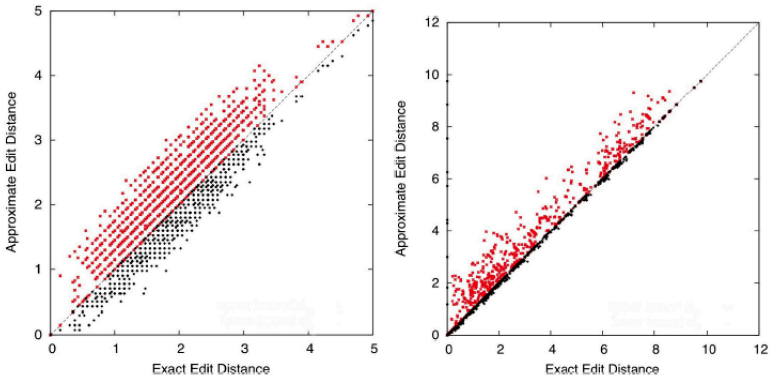
3. Expand the search tree for computing the edit distance between the following two graphs.



$[g_1]$ $[g_2]$

The nodes are labeled with integers, while the edges are unlabeled. You can use unit cost for deletions and insertions of both nodes and edges. Edge substitutions are free of cost, while the cost for substituting a node $u \in V_1$ with a node $v \in V_2$ is defined via $c(u \to v) = |\mu_1(u) - \mu_2(v)|$. Set $h(\lambda) = 0$ for all edit paths, i.e. use no heuristic information.

4. Consider the scatter plots in the figure below. They show the exact (x-axis) vs. the approximate (y-axis) graph edit distance computed with BP-GED resulting in d_ψ (red points) and d'_ψ (black points) on two different graph data sets. Elaborate on these figures (What can be observed?)



(a) AIDS (b) FP

Implementation Tasks

In this exercise series, the goal is to implement of the (lower or upper-bound) Bipartite-Graph Edit Distance (BP-GED) presented in the lecture notes. First, go to the ILIAS's webpage of the course and download/unzip `Exercise_2.zip` in your `PR_Lecture` folder. Then, navigate to `PR_lecture/Exercise_2/ex2.py` and complete the missing part of the source code.

1. Implement BP-GED algorithm in `Exercise_2/ex2.py`.
   (a) Compute the extended cost matrices $\mathbf{C}$ and $\mathbf{C}^*$. The costs are defined as follow:
      - $c(u \to u') = |\mu_1(u) - \mu_2(u')|$
      - $c(u \to \epsilon) = c(\epsilon \to u') = c((u, v) \to \epsilon) = c(\epsilon \to (u', v')) = \tau = 1$
      - $c((u, v) \to (u', v')) = 0$
   (b) Use the `scipy.optimize.linear_sum_assigment` as LSAP solver to find the optimal solution on $\mathbf{C}*$.
   (c) Derive the lower- or upper-bound approximated GED.
2. In the second part, you will compute BP-GED between all pairs of graphs in `Exercise_2/graphs`. To this end, create an $N \times N$ matrix $\mathbf{D} = (d_{ij})$, where N = is the number of graphs in the `PR_lecture/Exercise_2/graphs` directory. Matrix $\mathbf{D}$ contains the results of your BP-GED implementation for all pairs of graphs. More formally, entry d_{ij} at position i, j corresponds to the BP-GED between graph g_i and graph g_j.

   Save your matrix as `GED_results.csv` in the result folder.

Submission

For the coding part, you must submit a `.zip` file containing the following files. Additionally, include your solution for the theoretical tasks as `*.pdf` in the same `.zip` file.